

Patent Application
Docket No.: 47242-00027USC1
Customer Number 30223

APPLICATION FOR UNITED STATES LETTERS PATENT

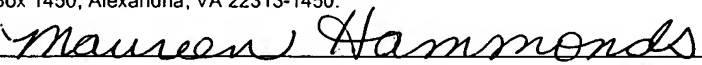
for

**ROBUST VOICE BROWSER SYSTEM AND
VOICE ACTIVATED DEVICE CONTROLLER**

INVENTORS:

Alexander Kurganov

Valery Zhukoff

EXPRESS MAIL MAILING LABEL	
NUMBER:	EV 284714429 US
DATE:	April 9, 2004
I hereby certify that this paper or fee is being deposited with the United States Postal Service "EXPRESS MAIL POST OFFICE TO ADDRESSEE" service under 37 C.F.R. 1.10 on the date indicated above and is addressed to: MS Patent Application, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.	
	

ROBUST VOICE BROWSER SYSTEM AND VOICE ACTIVATED DEVICE CONTROLLER

CROSS-REFERENCE TO RELATED APPLICATIONS

[001] This application is a continuation of U.S. Patent Application No. 09/776,996, filed February 5, 2001, now allowed, which claims priority to U.S. Provisional Application Serial No. 60/180,344, filed February 4, 2000 entitled "Voice-Activated Information Retrieval System" and U.S. Provisional Application Serial No. 60/233,068, filed September 15, 2000 entitled "Robust Voice Browser System and Voice Activated Device Controller" and these applications are incorporated by reference in their entirety into the present application.

FIELD OF THE INVENTION

[002] The present invention relates to a robust and highly reliable system that allows users to browse web sites and retrieve information by using conversational voice commands. Additionally, the present invention allows users to control and monitor other systems and devices that are connected the Internet or any other network by using voice commands.

BACKGROUND OF THE INVENTION

[003] Currently, three options exist for a user who wishes to gather information from a web site accessible over the Internet. The first option is to use a desktop or a laptop computer connected to a telephone line via a modem or connected to a network with Internet access. The second option is to use a Personal Digital Assistant (PDA) that has the capability of connecting to the Internet either through a modem or a wireless connection. The third option is to use one of the newly designed web-phones or web-pagers that are now being offered on the market. Although each of these options provide a user with access to the Internet to browse web sites, each of them have their own drawbacks.

[004] Desktop computers are very large and bulky and are difficult to transport. Laptop computers solve this inconvenience, but many are still quite heavy and are inconvenient to carry. Further, laptop computers cannot be carried and used everywhere a user travels. For instance, if a user wishes to obtain information from a remote location where no electricity or communication lines are installed, it would be nearly impossible to use a laptop computer. Oftentimes, information is needed on an immediate basis where a computer is

not accessible. Furthermore, the use of laptop or desktop computers to access the Internet requires either a direct or a dial-up connection tan an Internet Service Provider (ISP). Oftentimes, such connections are not available when a user desires to connect to the Internet to acquire information.

[005] The second option for remotely accessing web sites is the use of PDAs. These devices also have their own set of drawbacks. First, PDAs with the ability to connect to the Internet and access web sites are not readily available. As a result, these PDAs tend to be very expensive. Furthermore, users are usually required to pay a special service fee to enable the web browsing feature of the PDA. A further disadvantage of these PDAs is that web sites must be specifically designed to allow these devices to access information on the web site. Therefore, a limited number of web sites are available that are accessible by these web-enabled PDAs. Finally, it is very common today for users to carry cell phones, however, users must also carry a separate PDA if they require the ability to gather information from various web sites. Users are therefore subjected to added expenses since they must pay for both cellular telephone service and also for the web-enabling service for the PDA. This results in a very expensive alternative for the consumer.

[006] The third alternative mentioned above is the use of web-phones or web-pagers. These devices suffer many of the same drawbacks as PDAs. First, these devices are expensive to purchase. Further, the number of web sites accessible to these devices is limited since web sites must be specifically designed to allow access by these devices. Furthermore, users are often required to pay an additional fee in order to gain wireless web access. Again, this service is expensive. Another drawback of these web-phones or web-pagers is that as technology develops, the methods used by the various web sites to allow access by these devices may change. These changes may require users to purchase new web-phones or web-pagers or have the current device serviced in order to upgrade the firmware or operating system stored within the device. At the least, this would be inconvenient to users and may actually be quite expensive.

[007] Therefore, a need exists for a system that allows users to easily access and browse the Internet from any location. Such a system would only require users to have access to any type of telephone and would not require users to subscribe to multiple services.

[008] In the rapidly changing area of Internet applications, web sites change frequently. The design of the web site may change, the information required by the web site in order to

perform searches may change, and the method of reporting search results may change. Web browsing applications that submit search requests and interpret responses from these web sites based upon expected formats will produce errors and useless responses when such changes occur. Therefore, a need exists for a system that can detect modifications to web sites and adapt to such changes in order to quickly and accurately provide the information requested by a user through a voice enabled device, such as a telephone.

[009] When users access web sites using devices such as personal computers, delays in receiving responses are tolerated and are even expected, however, such delays are not expected when a user communicates with a telephone. Users expect communications over a telephone to occur immediately with a minimal amount of delay time. A user attempting to find information using a telephone expects immediate responses to his search requests. A system that introduces too much delay between the time a user makes a request and the time of response will not be tolerated by users and will lose its usefulness. Therefore, it is important that a voice browsing system that uses telephonic communications selects web sites that provide rapid responses since speed is an important factor for maintaining the system's desirability and usability. Therefore, a need exists for a system that accesses web sites based upon their speed of operation.

SUMMARY OF THE INVENTION

[0010] It is an object of an embodiment of the present invention to allow users to gather information from web sites by using voice enabled devices, such as wireline or wireless telephones.

[0011] An additional object of an embodiment of the present invention is to provide a system and method that allows the searching and retrieving of publicly available information by controlling a web browsing server using naturally spoken voice commands.

[0012] It is an object of another embodiment of the present invention to provide a robust voice browsing system that can obtain the same information from several web sites based upon a ranking order. The ranking order is automatically adjusted if the system detects that a given web site is not functioning, is too slow, or has been modified in such a way that the requested information cannot be retrieved any longer.

[0013] A still further object of an embodiment of the present invention is to allow users to gather information from web sites from any location where a telephonic connection can be

made.

[0014] Another object of an embodiment of the present invention is to allow users to browse web sites on the Internet using conversational voice commands spoken into wireless or wireline telephones or other voice enabled devices.

[0015] An additional object of an embodiment of the present invention is to provide a system and method for using voice commands to control and monitor devices connected to a network.

[0016] It is an object of an embodiment of the present invention to provide a system and method which allows devices connected to a network to be controlled by conversational voice commands spoken into any voice enabled device interconnected with the same network.

[0017] The present invention relates to a system for acquiring information from sources on a network, such as the Internet. A voice browsing system maintains a database containing a list of information sources, such as web sites, connected to a network. Each of the information sources is assigned a rank number which is listed in the database along with the record for the information source. In response to a speech command received from a user, a network interface system accesses the information source with the highest rank number in order to retrieve information requested by the user.

[0018] The preferred embodiment of the present invention allows users to access and browse web sites when they do not have access to computers with Internet access. This is accomplished by providing a voice browsing system and method that allows users to browse web sites using conversational voice commands spoken into any type of voice enabled device (i.e., any type of wireline or wireless telephone, IP phone, wireless PDA, or other wireless device). These spoken commands are then converted into data messages by a speech recognition software engine running on a user interface system. These data messages are then sent to and processed by a network interface system. This network interface system then generates the proper requests that are transmitted to the desired web site over the Internet. Responses sent from the web site are received and processed by the network interface system and then converted into an audio message via a speech synthesis engine or a pre-recorded audio concatenation application and finally transmitted to the user's voice enabled device.

[0019] A preferred embodiment of the voice browser system and method uses a web site

polling and ranking methodology that allows the system to detect changes in web sites and adapt to those changes in real-time. This enables the voice browser system of a preferred embodiment to deliver highly reliable information to users over any voice enabled device. This ranking system also enables the present invention to provide rapid responses to user requests. Long delays before receiving responses to requests are not tolerated by users of voice-based systems, such as telephones. When a user speaks into a telephone, an almost immediate response is expected. This expectation does not exist for non-voice communications, such as email transmissions or accessing a web site using a personal computer. In such situations, a reasonable amount of transmission delay is acceptable. The ranking system of implemented by a preferred embodiment of the present invention ensures users will always receive the fastest possible response to their request.

[0020] An alternative embodiment of the present invention allows users to control and monitor the operation of a variety of household devices connected to a network using speech commands spoken into a voice enabled device.

BRIEF DESCRIPTION OF THE DRAWINGS

[0021] FIG. 1 is a depiction of the voice browsing system of the first embodiment of the present invention;

[0022] FIG. 2 is a block diagram of a database record used by the first preferred embodiment of the present invention;

[0023] FIG. 3 is a block diagram of a media server used by the preferred embodiment;

[0024] FIG. 4 is a block diagram of a web browsing server used by the preferred embodiment; and

[0025] FIG. 5 is a depiction of the device browsing system of the second embodiment of the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

[0026] A first embodiment of the present invention is a system and method for allowing users to browse information sources, such as web sites, by using naturally spoken, conversational voice commands spoken into a voice enabled device. Users are not required to learn a

special language or command set in order to communicate with the voice browsing system of the present invention. Common and ordinary commands and phrases are all that is required for a user to operate the voice browsing system. The voice browsing system recognizes naturally spoken voice commands and is speaker-independent; it does not have to be trained to recognize the voice patterns of each individual user. Such speech recognition systems use phonemes to recognize spoken words and not predefined voice patterns.

[0027] The first embodiment allows users to select from various categories of information and to search those categories for desired data by using conversational voice commands. The voice browsing system of the first preferred embodiment includes a user interface system referred to as a media server. The media server contains a speech recognition software engine. This speech recognition engine is used to recognize natural, conversational voice commands spoken by the user and converts them into data messages based on the available recognition grammar. These data messages are then sent to a network interface system. In the first preferred embodiment, the network interface system is referred to as a web browsing server. The web browsing server then accesses the appropriate information source, such as a web site, to gather information requested by the user.

[0028] Responses received from the information sources are then transferred to the media server where speech synthesis engine converts the responses into audio messages that are transmitted to the user. A more detailed description of this embodiment will now be provided.

[0029] Referring to FIG. 1, a database 100 designed by Webley Systems Incorporated is connected to one or more web browsing servers 102 as well as to one or more media servers 106. The database may store information on magnetic media, such as a hard disk drive, or it may store information via other widely acceptable methods for storing data, such as optical disks. The database 100 contains a separate set of records for each web site accessible by the system. An example of a web site record is shown in FIG. 2. Each web site record 200 contains the rank number of the web site 202, the associated Uniform Resource Locator (URL) 204, and a command that enables the appropriate "extraction agent" 206 that is required in order to generate proper requests sent to and to format data received from the web site. The database record 200 also contains the timestamp 208

indicating the last time the web site was accessed. The extraction agent is described in more detail below. The database 100 categorizes each database record 200 according to the type of information provided by each web site. For instance, a first category of database records 200 may correspond to web sites that provide "weather" information. The database 100 may also contain a second category of records 200 for web sites that provide "stock" information. These categories may be further divided into subcategories. For instance, the "weather" category may contain subcategories depending upon type of weather information available to a user, such as "current weather" or "extended forecast". Within the "extended forecast" subcategory, a list of web site records may be stored that provide weather information for multiple days. The use of subcategories may allow the web browsing feature to provide more accurate, relevant, and up-to-date information to the user by accessing the most relevant web site. The number of records contained in each category or subcategory is not limited. In the preferred embodiment, three web site records are provided for each category.

[0030] Table 1 below depicts two database records 200 that are used with the preferred embodiment. These records also contain a field indicating the "category" of the record, which is "weather" in each of these examples.

TABLE 1

```
category:    weather
URL:    URL=http://cgi.cnn.com/cgi-
bin/weather/redirect?zip=_zip
rank:    1
command:    web_dispatch.pl weather_cnn
browsingServer:    wportal1
browsingServerBackup:    wportal2
dateTime:    Dec 21 2000  2:15PM
```

```
category:    weather
URL:    URL=http://weather.lycos.com/wcfiveday.asp?city=zip
rank:    2
command:    web_dispatch.pl weather_lycos
browsingServer:    wportal1
browsingServerBackup:    wportal2
dateTime:    Dec 21 2000  1:45PM
```

[0031] The database also contains a listing of pre-recorded audio files used to create

concatenated phrases and sentences. Further, database 100 may contain customer profile information, system activity reports, and any other data or software servers necessary for the testing or administration of the voice browsing system.

[0032] The operation of the media servers 106 will now be discussed in relation to FIG.

3. The media servers 106 function as user interface systems. In the preferred embodiment, the media servers 106 contain a speech recognition engine 300, a speech synthesis engine 302, an Interactive Voice Response (IVR) application 304, a call processing system 306, and telephony and voice hardware 308 required to communicate with the Public Switched Telephone Network (PSTN) 116. In the preferred embodiment, each media server is based upon Intel's Dual Pentium III 730 MHz microprocessor system.

[0033] The speech recognition function is performed by a speech recognition engine 300 that converts voice commands received from the user's voice enabled device 112 (i.e., any type of wireline or wireless telephone, Internet Protocol (IP) phones, or other special wireless units) into data messages. In the preferred embodiment, voice commands and audio messages are transmitted using the PSTN 116 and data is transmitted using the TCP/IP communications protocol. However, one skilled in the art would recognize that other transmission protocols may be used for either voice or data. Other possible transmission protocols would include SIP/VoIP (Session Initiation Protocol/Voice over IP), Asynchronous Transfer Mode (ATM) and Frame Relay. A preferred speech recognition engine is developed by Nuance Communications of 1380 Willow Road, Menlo Park, California 94025 (www.nuance.com). The Nuance engine capacity is measured in recognition units based on CPU type as defined in the vendor specification. The natural speech recognition grammars (i.e., what a user can say that will be recognized by the speech recognition engine) were developed by Webley Systems.

[0034] Table 2 below provides a partial source code listing of the recognition grammars used by the speech recognition engine of the preferred embodiment for obtaining weather information.

TABLE 2

```
?WHAT_IS ?the weather ?[info information report conditions]
      ? ( (?like in )
      [
```

UScities:n

```
{<param1 $n.zip> <param2 $n.city> <param3
$n.state>}
( (area code) AREA_CODE:n ) {<param1 $n>}
( AREA_CODE:n (area code) ) {<param1 $n>}
( (zip ?code) ZIP_CODE:n ) {<param1 $n>}
( ZIP_CODE:n (zip ?code) ) {<param1 $n>}
]
)
) {<menu 194>}
```

[0035] The media server 106 uses recognition results generated by the speech recognition engine 300 to retrieve a web site record 200 stored in the database 100 that can provide the information requested by the user. The media server 106 processes the recognition result data identifying keywords that are used to search the web site records 200 contained in the database 100. For instance, if the user's request was "What is the weather in Chicago?", the keywords "weather" and "Chicago" would be recognized. A web site record 200 with the highest rank number from the "weather" category within the database 100 would then be selected and transmitted to the web browsing server 102 along with an identifier indicating that Chicago weather is being requested.

[0036] The media servers 106 also contain a speech synthesis engine 302 that converts the data retrieved by the web browsing servers 102 into audio messages that are transmitted to the user's voice enabled device 112. A preferred speech synthesis engine is developed by Lernout and Hauspie Speech Products, 52 Third Avenue, Burlington, Massachusetts 01803 (www.lhsl.com).

[0037] A further description of the web browsing server 102 will be provided in relation to FIG. 4. The web browsing servers 102 provide access to any computer network such as the Internet 110. These servers are also capable of accessing databases stored on Local Area Networks (LANs) or Wide Area Networks (WANs). The web browsing servers receive responses from web sites and extract the data requested by the user. This task is also known as "content extraction." The web browsing servers 102 also perform the task of periodically polling or "pinging" various web sites and modifying the ranking numbers of these web sites depending upon their response and speed. This polling feature is further discussed below. The web browsing server 102 is comprised of a content

extraction agent 400, a content fetcher 402, a polling and ranking agent 404, and the content descriptor files 406. Each of these are software applications and will be discussed below.

[0038] Upon receiving a web site record 200 from the database 100 in response to a user request, the web browsing server 102 invokes the "content extraction agent" command 206 contained in the record 200. The content extraction agent 400 allows the web browsing server 102 to properly format requests and read responses provided by the web site 114 identified in the URL field 204 of the web site record 200. Each content extraction agent command 206 invokes the content extraction agent and identifies a content description file associated with the web page identified by the URL 204. This content description file directs the extraction agent where to extract data from the accessed web page and how to format a response to the user utilizing that data. For example, the content description for a web page providing weather information would indicate where to insert the "city" name or ZIP code in order to retrieve Chicago weather information. Additionally, the content description file for each supported URL indicates the location on the web page where the response information is provided. The extraction agent 400 uses this information to properly extract from the web page the information requested by the user.

[0039] Table 3 below contains source code for a content extraction agent 400 used by the preferred embodiment.

TABLE 3

```
#!/usr/local/www/bin/sybperl5
#$Header:
/usr/local/cvsroot/webley/agents/service/web_dispatch.pl,v
1.6
# Dispatches all web requests

#http://wcorp.itn.net/cgi/flstat?carrier=ua&flight_no=155&mon_
abbr=jul&date=
6&stamp=OhLN~PdbuuE*itn/ord,itn/cb/sprint_hd
#http://cgi.cnnfn.com/flightview/rlm?airline=amt&number=300

require "config_tmp.pl";

# check parameters
die "Usage: $0 service [params]\n" if $#ARGV < 1;
#print STDERR @ARGV;
```

```

# get parameters
my ( $service, @param ) = @ARGV;

# check service
my %Services = (
    weather_cnn => 'webget.pl weather_cnn',
    weather_lycos => 'webget.pl
weather_lycos',
    weather_weather => 'webget.pl
weather_weather',
    weather_snap => 'webget.pl
weather_snap',
    weather_infospace => 'webget.pl
weather_infospace',
    stockQuote_yahoo => 'webget.pl stock',
    flightStatus_itn => 'webget.pl
flight_delay',
    yellowPages_yahoo => 'yp_data.pl',
    yellowPages_yahoo => 'yp_data.pl',
    newsHeaders_newsreal => 'news.pl',
    newsArticle_newsreal => 'news.pl',
);

# test param
my $date = 'date';
chop( $date );
my ( $short_date ) = $date =~ /\s+(\w{3})\s+\d{1,2})\s+;/;
my %Test = (
    weather_cnn => '60053',
    weather_lycos => '60053',
    weather_weather => '60053',
    weather_snap => '60053',
    weather_infospace => '60053',
    stockQuote_yahoo => 'msft',
    flightStatus_itn => 'ua 155 ' .
$short_date,
    yellowPages_yahoo => 'tires 60015',
    newsHeaders_newsreal => '1',
    newsArticle_newsreal => '1 1',
);

die "$date: $0: error: no such service: $service (check this
script)\n"
unless $Services{$service };

# prepare absolute path to run other scripts
my ( $path, $script ) = $0 =~ m|^^(.*)\|^/.*$|;

# store the service to compare against datatable
my $service_stored = $service;

```

```
# run service
while( !( $response = '$path$Services{$service }@param' ) )
{
    # response failed
    # check with test parameters
    $response = '$path$Services{$service }$Test{$service
}';
    # print "test: $path$Services{$service }$Test{$service
}";
    if ( $response ) {
        $service = &switch_service( $service );
        # print "Wrong parameter values were supplied:
$service -
@param\n";
        # die "$date: $0: error: wrong parameters: $service
-
@param\n";
    }
    else {
        # change priority and notify
        $service = &increase_attempt( $service );
    }
}

# output the response :
print $response;

sub increase_attempt {
    my ( $service ) = @_ ;
    my ( $service_name ) = split( /\_/, $service );
    print STDERR "$date: $0: attn: changing priority for
service:
$service\n";
    # update priority
    &db_query( "update mcServiceRoute "
                . "set priority = ( select max( priority )
from
mcServiceRoute "
                . "where service = '$service_name' ) + 1,
"
                . "date = getdate(), "
                . "attempt = attempt + 1 "
                . "where route = '$script $service'" );
    # print "---$route===\n";
    # find new route
    my $route = @{&db_query( "select route from
mcServiceRoute "
                            . "where service =
'$service_name' "
                            . "and attempt < 5
```

```

"
. "order by
priority ")
    }-> [ 0 ]{route };
    &db_query( "update mcServiceRoute "
        . "set attempt = 0 "
        . "where route = '$script $service'" )
    if ( $route eq "$script $service"
        or $route eq "$script $service_stored" );
    ( $service_name, $service ) = split( /\s+/, $route );
    die "$date: $0: error: no route for the service:
$service (add
more)\n"
        unless $service;
    return $service;
}
sub switch_service {
    my ( $service ) = @_;
    my ( $service_name ) = split( /\s+/, $service );
    print STDERR "$date: $0: attn: changing priority for
service:
$service\n";
    # update priority
    &db_query( "update mcServiceRoute "
        . "set priority = ( select max( priority )
from
mcServiceRoute "
        . "where service = '$service_name' ) + 1,
"
        . "date = getdate() "
        . "where route = '$script $service'" );
    # print "---$route===\n";
    # find new route
    my $route = @{&db_query( "select route from
mcServiceRoute "
        . "where service =
'$service_name' "
        . "and attempt < 5
"
        . "order by
priority ")
    }-> [ 0 ]{route };
    die "$date: $0: error: there is the only service:
$route (add
more)\n"
        if ( $route eq "$script $service"
            or $route eq "$script $service_stored" );
    ( $service_name, $service ) = split( /\s+/, $route );
    die "$date: $0: error: no route for the service:
$service (add

```

```

more)\n"
        unless $service;
        return $service;
    }
    _

```

[0074] Table 4 below contains source code of the content fetcher 402 used with the content extraction agent 400 to retrieve information from a web site.

TABLE 4

```

#!/usr/local/www/bin/sybperl5
#-T
# -w
# $Header:
# /usr/local/cvsroot/webley/agents/service/webget.pl,v 1.4
# Agent to get info from the web.
# Parameters: service_name [service_parameters], i.e. stock
# msft or weather
60645
# Configuration stored in files service_name.ini
# if this file is absent the configuration is received from
# mcServices table
# This script provides autoupdate to datatable if the .ini
# file is newer.

$debug = 1;
use URI::URL;
use LWP::UserAgent;
use HTTP::Request::Common;
use Vail::VarList;
use Sybase::CTlib;
use HTTP::Cookies;
#print "Sybase::CTlib $DB_USR, $DB_PWD, $DB_SRV;";
open( STDERR, ">>$0.log" ) if $debug;
#open( STDERR, ">&STDOUT" );
$log = 'date';
$response = './url.pl
"http://cgi.cnn.com/cgi-bin/weather/redirect?zip=60605";
$response = 'pwd';
#print STDERR "pwd = $response\n";
$response = 'ls';
#print STDERR "ls = $response\n";
chop( $log );
$log .= "pwd=" . 'pwd';
chop( $log );

#$debug2 = 1;

```

```

my $service = shift;
$log .= " $service: " . join( ':', @ARGV ) . "\n";
print STDERR $log if $debug;
#$response = `./url.pl
"http://cgi.cnn.com/cgi-bin/weather/redirect?zip=60605"`;
my @ini = &read_ini( $service );
chop( @ini );
my $section = "";
do { $section = &process_section( $section ) } while $section;
#$response = `./url.pl
"http://cgi.cnn.com/cgi-bin/weather/redirect?zip=60605"`;
exit;
#####
sub read_ini {
    my ( $service ) = @_;
    my @ini = ();
    # first, try to read file
    $0 =~ m|^(\./)[^/]*|;
    $service = $1 . $service;
    if ( open( INI, "$service.ini" ) ) {
        @ini = ( <INI> );
        return @ini unless ( $DB_SRV );
        # update datatable
        my $file_time = time - int( ( -M "$service.ini" )
* 24 *
3600 );
        #
        print "time $file_time\n";
        my $dbh = new Sybase::CTlib $DB_USR, $DB_PWD,
$DB_SRV;
        unless ( $dbh ) {
            print STDERR "webget.pl: Cannot connect to
dataserver $DB_SRV:$DB_USR:$DB_PWD\n";
            return @ini;
        }
        my @row_refs = $dbh->ct_sql( "select lastUpdate
from
mcServices where service = '$service'", undef, 1 );
        if ( $dbh->{RC} == CS_FAIL ) {
            print STDERR "webget.pl: DB select from
mcServices
failed\n";
            return @ini;
        }
        unless ( defined @row_refs ) {
            # have to insert
            my ( @ini_escaped ) = map {
                ( my $x = $_ ) =~ s/\\/\\/\\/\\/g;
                $x;
            } @ini;
            $dbh->ct_sql( "insert mcServices values(

```



```

'$service',
'@ini_escaped', $file_time )" );
    if ( $dbh->{RC }== CS_FAIL ) {
        print STDERR "webget.pl: DB insert to
mcServices failed\n";
    }
    return @ini;
}
#      print "time $file_time:". $row_refs[ 0 ]-
>{'lastUpdate'
}." \n";
    if ( $file_time > $row_refs[ 0 ]->{'lastUpdate' })
{
    # have to update
    my ( @ini_escaped ) = map {
        ( my $x = $_ ) =~ s/\'/\\'/g;
        $x;
    }@ini;
    $dbh->ct_sql( "update mcServices set config =
'@ini_escaped', lastUpdate = $file_time where service =
'$service'" );
    if ( $dbh->{RC }== CS_FAIL ) {
        print STDERR "webget.pl: DB update to
mcServices failed\n";
    }
    return @ini;
}
else {
    print STDERR "$0: WARNING: $service.ini n/a in " .
'pwd'
        . "Try to read DB\n";
}
# then try to read datatable
die "webget.pl: Unable to find service $service\n"
unless ( $DB_SRV
);
my $dbh = new Sybase::CTlib $DB_USR, $DB_PWD, $DB_SRV;
die "webget.pl: Cannot connect to dataserver
$DB_SRV:$DB_USR:$DB_PWD\n" unless ( $dbh );
my @row_refs = $dbh->ct_sql( "select config from
mcServices where
service = '$service'", undef, 1 );
die "webget.pl: DB select from mcServices failed\n" if
$dbh->{RC }
== CS_FAIL;
die "webget.pl: Unable to find service $service\n"
unless ( defined
@row_refs );
$row_refs[ 0 ]->{'config' } =~ s/\n /\n\r/g;

```

```

@ini = split( /\r/, $row_refs[ 0 ]->{'config' } );
return @ini;
}
#####
sub process_section {
    my ( $prev_section ) = @_ ;
    my ( $section, $output, $content );
    my %Param;
    my %Content;
#   print "#####\n";
    foreach ( @ini ) {
#       print;
#       chop;
        s/\s+$/;/;
        s/^\s+//;
        # get section name
        if ( /^\[([.]*)\]/ ) {
#           print "$_: $section:$prev_section\n";
            last if $section;
            next if $1 eq "print";
#           next if $prev_section ne "" and $prev_section
ne $1;

            if ( $prev_section eq $1 ) {
                $prev_section = "";
                next;
            }
            $section = $1;
        }
        # get parameters
        push( @{$Param{$1}}, $2 ) if $section and
/([^\s]+)=(.*)/;
    }
#   print "#####\n";
    return 0 unless $section;
#   print "section $section\n";
    # substitute parameters with values
    map { $Param{URL }->[ 0 ] =~ s/$Param{Input }->[ $
]/$ARGV[ $_
]/g
        } 0 .. $#{$Param{Input }};

    # get page content
    ( $Content{'TIME' }, $content ) = &get_url_content(
${$Param{URL
}}[ 0 ] );

    # filter it
    map {
        if ( /\\"([^\"]+)\\"([^\"]*)\\"/ or
/\\"([^\"]+)\\"([^\"]*)\\"/ )

```

```

{
    my $out = $2; $content =~ s/$1/$out/g;
}
} @{$Param{"Pre-filter" }};

#print STDERR $content;
# do main regular expression
unless( @values = $content =~
/${$Param{Regular_expression }}[ 0
]/ ) {
    &die_hard( ${$Param{Regular_expression }}[ 0 ],
$content
);
    return $section;
}

%Content = map { ( $Param{Output }->[ $_ ], $values[ $_
] )
} 0 .. ${$Param{Output }};

# filter it
map {
    if ( /(^[^"]+)\\"([^\"]+)\\"([^\"]*)\\"/
        or /(^[^/]+\\"([^\"]+)\\"([^\"]*)\\"// ) {
        my $out = $3;
        $Content{$1 } =~ s/$2/$out/g;
    }
} @{$Param{"Post-filter" }};

# calculate it
map {
    if ( /(^[^=]+)=(.*)/ ) {
        my $eval = $2;
        map { $eval =~ s/$_/$Content{$_ }/g
        } keys %Content;
        $Content{$1 } = eval( $eval );
    }
} @{$Param{Calculate }};

# read section [print]
foreach $i ( 0 .. $#ini ) {
    next unless $ini[ $i ] =~ /^\[print\]/;
    foreach ( $i + 1 .. $#ini ) {
        last if $ini[ $_ ] =~ /^\[.+\]/;
        $output .= $ini[ $_ ] . "\n";
    }
    last;
}
}

```

```

# prepare output
map {$output =~ s/$_/$Content{$_ }/g
}keys %Content;
print $output;
return 0;
}
#####
sub get_url_content {
    my ( $url ) = @_ ;
    print STDERR $url if $debug;
#   $response = `./url.pl '$url'`;
    $response = `./url.pl '$url'`;
    return( $time - time, $response );
    my $ua = LWP::UserAgent->new;
    $ua->agent( 'Mozilla/4.0 [en] (X11; I; FreeBSD 2.2.8-
STABLE i386)'
);
#   $ua->proxy( ['http', 'https'],
'http://proxy.webley:3128/' );
#   $ua->no_proxy( 'webley', 'vail' );
    my $cookie = HTTP::Cookies->new;
    $ua->cookie_jar( $cookie );
    $url = url $url;
    print "$url\n" if $debug2;
    my $time = time;
    my $res = $ua->request( GET $url );
    print "Response: " . ( time - $time ) . "sec\n" if
$debug2;
    return( $time - time, $res->content );
}
#####
sub die_hard {
    my( $re, $content ) = @_ ;
    my ( $re_end, $pattern );
    while( $content !~ /$re/ ) {
        if ( $re =~ s/(\([^(\)]+\)[^\(\)]*$)// ) {
            $re_end = $1 . $re_end;
        }
        else {
            $re_end = $re;
            last;
        }
    }
    $content =~ /$re/;
    print STDERR "The regular expression did not match:\n
$re\n
Possible misuse:
$re_end:\n
Matched:
$&\n

```

```

Mismatched:
$'\n
" if $debug;
    if ( $debug ) {
        print STDERR "Content:\n $content\n" unless
$';
    }
}

```

#####

[0075] Table 5 below contains the content descriptor file source code for obtaining weather information from the web site www.cnn.com that is used by the extraction agent 400 of the preferred embodiment.

TABLE 5

```

[cnn]
Input=_zip

URL=http://cgi.cnn.com/cgi-bin/weather/redirect?zip=_zip

Pre-filter="\n"
Pre-filter="<[^<>]+>"
Pre-filter="/s+/ /
Pre-filter="\(\)\|\]"

Output=_location
Output=first_day_name
Output=first_day_weather
Output=first_day_high_F
Output=first_day_high_C
Output=first_day_low_F
Output=first_day_low_C
Output=second_day_name
Output=second_day_weather
Output=second_day_high_F
Output=second_day_high_C
Output=second_day_low_F
Output=second_day_low_C
Output=third_day_name
Output=third_day_weather
Output=third_day_high_F
Output=third_day_high_C
Output=third_day_low_F
Output=third_day_low_C
Output=fourth_day_name
Output=fourth_day_weather

```

Output=fourth_day_high_F
Output=fourth_day_high_C
Output=fourth_day_low_F
Output=fourth_day_low_C
Output=undef
Output=_current_time
Output=_current_month
Output=_current_day
Output=_current_weather
Output=_current_temperature_F
Output=_current_temperature_C
Output=_humidity
Output=_wind
Output=_pressure
Output=_sunrise
Output=_sunset

Regular_expression=Author (.) Four Day Forecast
(\S+) (\S+) HIGH
(\S+) F (\S+) C LOW (\S+) F (\S+) C (\S+) (\S+) HIGH (\S+) F
(\S+) C LOW
(\S+) F (\S+) C (\S+) (\S+) HIGH (\S+) F (\S+) C LOW (\S+) F
(\S+) C (\S+)
(\S+) HIGH (\S+) F (\S+) C LOW (\S+) F (\S+) C (.) Current
Conditions(.)
!local!, (\S+) (\S+) (.) Temp: (\S+) F, (\S+) C Rel.
Humidity: (\S+) Wind:
(.) Pressure: (.) Sunrise: (.) Sunset: (.) Related Links

Post-filter=_current_weather"p/"partly "
Post-filter=_current_weather"l/"little "
Post-filter=_current_weather"m/"mostly "
Post-filter=_current_weather"t-/"thunder "
Post-filter=_wind"N"North "
Post-filter=_wind"E"East "
Post-filter=_wind"S"South "
Post-filter=_wind"W"West "
Post-filter=_wind/mpH/miles per hour/
Post-filter=_wind/kph!/kilometers per hour/
Post-filter=_wind"\s+!", "

[print]
Current weather in _location is _current_weather.
Temperature is _current_temperature_F Fahrenheit,
_current_temperature_C
Celsius.
Humidity is _humidity.
Wind from the _wind.

[0076] Table 6 below contains the content descriptor file source code for obtaining weather information from the web site www.lycos.com that is used by the extraction agent 400 of the preferred embodiment.

TABLE 6

[lycos]

Input=zip

Input=_city

URL=<http://weather.lycos.com/wcfiveday.asp?city=zip>

Pre-filter="\n" "

Pre-filter="</TD>"td"

Pre-filter="<!.*?->"

Pre-filter="
"_br_"

Pre-filter="/alt=""/>alt="/

Pre-filter="<[^<>]+>"

Pre-filter=" " "

Pre-filter="/s+/ /

Output=_location

Output=_current_weather

Output=_current_temperature_F

Output=_humidity

Output=_winddir

Output=_windspeed

Output=_windmeasure

Output=_pressure

Output=first_day_name

Output=second_day_name

Output=third_day_name

Output=fourth_day_name

Output=fifth_day_name

Output=first_day_weather

Output=second_day_weather

Output=third_day_weather

Output=fourth_day_weather

Output=fifth_day_weather

Output=first_day_high_F

Output=first_day_low_F

Output=second_day_high_F

Output=second_day_low_F

Output=third_day_high_F

Output=third_day_low_F

Output=fourth_day_high_F

Output=fourth_day_low_F

```
Output=fifth_day_high_F
Output=fifth_day_low_F
Output=_windkmh
```

```
Regular_expression=Guide My Lycos (.) Click image to
enlarge
alt=("[^"]+)"(?:.+) Temp: (\d+)(?:.+)F _br_ Humidity:
(\S+)(?:.+) Wind: (.*?)
_br_
```

```
Output=_current_temperature_C
```

```
Post-filter=_location"_br_"
Post-filter=_current_weather"p/"partly "
Post-filter=_current_weather"m/"mostly "
Post-filter=_current_weather"t-/"thunder "
Post-filter=_winddir"@ " at "
Post-filter=_winddir/mph/miles per hour/
Post-filter=_wind/kph!/kilometers per hour/
```

```
Calculate=_current_temperature_C=int((_current_temperature_F
-32)*5/9)
```

```
Calculate=_windkmh=int(_windspeed*1.6)
```

```
[print]
The current weather in _location is _current_weather.
The current temperature is _current_temperature_F Fahrenheit
_current_temperature_C Celcius.
Humidity is _humidity.
```

```
Winds _winddir.
```

[0077] Once the web browsing server 102 accesses the web site specified in the URL 204 and retrieves the requested information, the information is forwarded to the media server 106. The media server uses the speech synthesis engine 302 to create an audio message that is then transmitted to the user's voice enabled device 112. In the preferred embodiment, each web browsing server 102 is based upon Intel's Dual Pentium III 730 MHz microprocessor system.

[0078] Referring to FIG. 1, the operation of the robust voice browser system will be described. A user establishes a connection between his voice enabled device 112 and a media server 106. This may be done using the Public Switched Telephone Network

(PSTN) 116 by calling a telephone number associated with the voice browsing system 118. Once the connection is established, the media server 106 initiates an interactive voice response (IVR) application 304. The IVR application plays audio messages to the user presenting a list of options, such as, "stock quotes", "flight status", "yellow pages", "weather", and "news". These options are based upon the available web site categories and may be modified as desired. The user selects the desired option by speaking the name of the option into the voice enabled device 112.

[0079] As an example, if a user wishes to obtain restaurant information, he may speak into his telephone the phrase "yellow pages". The IVR application would then ask the user what he would like to find and the user may respond by stating "restaurants". The user may then be provided with further options related to searching for the desired restaurant. For instance, the user may be provided with the following restaurant options, "Mexican Restaurants", "Italian Restaurants", or "American Restaurants". The user then speaks into the telephone 112 the restaurant type of interest. The IVR application running on the media server 106 may also request additional information limiting the geographic scope of the restaurants to be reported to the user. For instance, the IVR application may ask the user to identify the zip code of the area where the restaurant should be located. The media server 106 uses the speech recognition engine 300 to interpret the speech commands received from the user. Based upon these commands, the media server 106 retrieves the appropriate web site record 200 from the database 100. This record and any additional data, which may include other necessary parameters needed to perform the user's request, are transmitted to a web browsing server 102. A firewall 104 may be provided that separates the web browsing server 102 from the database 100 and media server 106. The firewall provides protection to the media server and database by preventing unauthorized access in the event the firewall for web browsing server 108 fails or is compromised. Any type of firewall protection technique commonly known to one skilled in the art could be used, including packet filter, proxy server, application gateway, or circuit-level gateway techniques.

[0080] The web browsing server 102 then uses the web site record and any additional data and executes the extraction agent 400 and relevant content descriptor file 406 to retrieve the requested information.

[0081] The information received from the responding web site 114 is then processed by

the web browsing server 102 according to the content descriptor file 406 retrieval by the extraction agent. This processed response is then transmitted to the media server 106 for conversion into audio messages using either the speech synthesis software 302 or selecting among a database of prerecorded voice responses contained within the database 100.

[0082] As mentioned above, each web site record contains a rank number 202 as shown in FIG. 2. For each category searchable by a user, the database 100 may list several web sites, each with a different rank number 202. As an example, three different web sites may be listed as searchable under the category of "restaurants". Each of those web sites will be assigned a rank number such as 1, 2, or 3. The site with the highest rank (i.e., rank = 1) will be the first web site accessed by a web browsing server 102. If the information requested by the user cannot be found at this first web site, then the web browsing server 102 will search the second ranked web site and so forth down the line until the requested information is retrieved or no more web sites left to check.

[0083] The web site ranking method and system of the present invention provides robustness to the voice browser system and enables it to adapt to changes that may occur as web sites evolve. For instance, the information required by a web site 114 to perform a search or the format of the reported response data may change. Without the ability to adequately monitor and detect these changes, a search requested by a user may provide an incomplete response, no response, or an error. Such useless responses may result from incomplete data being provided to the web site 114 or the web browsing server 102 being unable to recognize the response data messages received from the searched web site 114.

[0084] The robustness and reliability of the voice browsing system of the present invention is further improved by the addition of a polling mechanism. This polling mechanism continually polls or "pings" each of the sites listed in the database 100. During this polling function, a web browsing server 102 sends brief data requests or "polling digital data" to each web site listed in database 100. The web browsing server 102 monitors the response received from each web site and determines whether it is a complete response and whether the response is in the expected format specified by the content descriptor file 406 used by the extraction agent 400. The polled web sites that provide complete responses in the format expected by the extraction agent 400 have their ranking established based on their "response time". That is, web sites with faster

response times will be assigned higher rankings than those with slower response times. If the web browsing server 102 receives no response from the polled web site or if the response received is not in the expected format, then the rank of that web site is lowered. Additionally, the web browsing server contains a warning mechanism that generates a warning message or alarm for the system administrator indicating that the specified web site has been modified or is not responsive and requires further review.

[0085] Since the web browsing servers 102 access web sites based upon their ranking number, only those web sites that produce useful and error-free responses will be used by the voice browser system to gather information requested by the user. Further, since the ranking numbers are also based upon the speed of a web site in providing responses, only the most time efficient sites are accessed. This system assures that users will get complete, timely, and relevant responses to their requests. Without this feature, users may be provided with information that is not relevant to their request or may not get any information at all. The constant polling and re-ranking of the web sites used within each category allows the voice browser of the present invention to operate efficiently. Finally, it allows the voice browser system of the present invention to dynamically adapt to changes in the rapidly evolving web sites that exist on the Internet.

[0086] It should be noted that the web sites accessible by the voice browser of the preferred embodiment may use any type of mark-up language, including Extensible Markup Language (XML), Wireless Markup Language (WML), Handheld Device Markup Language (HDML), Hyper Text Markup Language (HTML), or any variation of these languages.

[0087] A second embodiment of the present invention is depicted in FIG. 5. This embodiment provides a system and method for controlling a variety of devices 500 connected to a network 502 by using conversational speech commands spoken into a voice enabled device 504 (i.e., wireline or wireless telephones, Internet Protocol (IP) phones, or other special wireless units). The networked devices may include various household devices. For instance, voice commands may be used to control household security systems, VCRs, TVs, outdoor or indoor lighting, sprinklers, or heating and air conditioning systems.

[0088] Each of these devices 500 is connected to a network 502. These devices 500 may contain embedded microprocessors or may be connected to other computer equipment

that allow the device 500 to communicate with network 502. In the preferred embodiment, the devices 500 appear as "web sites" connected to the network 502. This allows a network interface system, such as a device browsing server 506, a database 508, and a user interface system, such as a media server 510, to operate similar to the web browsing server 102, database 100 and media server 106 described in the first preferred embodiment above. A network 502 interfaces with one or more network interface systems, which are shown as device browsing servers 506 in FIG. 5. The device browsing servers perform many of the same functions and operate in much the same way as the web browsing servers 102 discussed above in the first preferred embodiment. The device browsing servers 506 are also connected to a database 508.

[0089] Database 508 lists all devices that are connected to the network 502. For each device 500, the database 508 contains a record similar to that shown in FIG. 2. Each record will contain at least a device identifier, which may be in the form of a URL, and a command to "content extraction agent" contained in the device browsing server 506. Database 508 may also include any other data or software necessary to test and administer the device browsing system.

[0090] The content extraction agent operates similarly to that described in the first embodiment. A device descriptor file contains a listing of the options and functions available for each of the devices 500 connected on the network 502. Furthermore, the device descriptor file contains the information necessary to properly communicate with the networked devices 500. Such information would include, for example, communication protocols, message formatting requirements, and required operating parameters.

[0091] The device browsing server 506 receives messages from the various networked devices 500, appropriately formats those messages and transmits them to one or more media servers 510 which are part of the device browsing system. The user's voice enabled devices 504 can access the device browsing system by calling into a media server 510 via the Public Switched Telephone Network (PSTN) 512. In the preferred embodiment, the device browsing server is based upon Intel's Dual Pentium III 730 MHz microprocessor system.

[0092] The media servers 510 act as user interface systems and perform the functions of natural speech recognition, speech synthesis, data processing, and call handling. The

media server 510 operates similarly to the media server 106 depicted in FIG. 3. When data is received from the device browser server 506, the media server 510 will convert the data into audio messages via a speech synthesis engine that are then transmitted to the voice enabled device of the user 504. Speech commands received from the voice enabled device of the user 504 are converted into data messages via a speech recognition engine running on the media server 510. A preferred speech recognition engine is developed by Nuance Communications of 1380 Willow Road, Menlo Park, California 94025 (www.nuance.com). A preferred speech synthesis engine is developed by Lernout and Hauspie Speech Products, 52 Third Avenue, Burlington, Massachusetts 01803 (www.lhsl.com). The media servers 510 of the preferred embodiment are based on Intel's Dual Pentium III 730 MHz microprocessor system. A specific example for using the system and method of this embodiment of the invention will now be given.

[0093] First, a user may call into a media server 510 by dialing a telephone number associated with an established device browsing system. Once the user is connected, the IUR application of the media server 510 will provide the user with a list of available systems that may be monitored or controlled based upon information contained in database 508.

[0094] For example, the user may be provided with the option to select "Home Systems" or "Office Systems". The user may then speak the command "access home systems". The media server 510 would then access the database 508 and provide the user with a listing of the home subsystems or devices 500 available on the network 502 for the user to monitor and control. For instance, the user may be given a listing of subsystems such as "Outdoor Lighting System", "Indoor Lighting System", "Security System", or "Heating and Air Conditioning System". The user may then select the indoor lighting subsystem by speaking the command "Indoor Lighting System". The IUR application would then provide the user with a set of options related to the indoor lighting system. For instance the media server 510 may then provide a listing such as "Dining Room", "Living Room", "Kitchen", or "Bedroom". After selecting the desired room, the IUR application would provide the user with the options to hear the "status" of the lighting in that room or to "turn on", "turn off", or "dim" the lighting in the desired room. These commands are provided by the user by speaking the desired command into the users voice enabled device 504. The media server 510 receives this command and translates it into a data

message. This data message is then forwarded to the device browsing server 506 which routes the message to the appropriate device 500.

[0095] The device browsing system 514 of this embodiment of the present invention also provides the same robustness and reliability features described in the first embodiment. The device browsing system 514 has the ability to detect whether new devices have been added to the system or whether current devices are out-of-service. This robustness is achieved by periodically polling or "pinging" all devices 500 listed in database 508. The device browsing server 506 periodically polls each device 500 and monitors the response. If the device browsing server 506 receives a recognized and expected response from the polled device, then the device is categorized as being recognized and in-service. However, if the device browsing server 506 does not receive a response from the polled device 500 or receives an unexpected response, then the device 500 is marked as being either new or out-of-service. A warning message or a report may then be generated for the user indicating that a new device has been detected or that an existing device is experiencing trouble.

[0096] Therefore, this embodiment allows users to remotely monitor and control any devices that are connected to a network, such as devices within a home or office. Furthermore, no special telecommunications equipment is required for users to remotely access the device browser system. Users may use any type of voice enabled device (i.e., wireline or wireless telephones, IP phones, or other wireless units) available to them. Furthermore, a user may perform these functions from anywhere without having to subscribe to additional services. Therefore, no additional expenses are incurred by the user.

[0097] The descriptions of the preferred embodiments described above are set forth for illustrative purposes and are not intended to limit the present invention in any manner. Equivalent approaches are intended to be included within the scope of the present invention. While the present invention has been described with reference to the particular embodiments illustrated, those skilled in the art will recognize that many changes and variations may be made thereto without departing from the spirit and scope of the present invention. These embodiments and obvious variations thereof are contemplated as falling within the scope and spirit of the claimed invention.